EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L2	26	(717/151-159.ccls.) and (optimiz$5 and check$1point)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2006/11/13 14:37
L3	27	(717/124-131.ccls.) and (optimiz$5 and check$1point)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2006/11/13 14:37
S1	48	("6658656" "20040034814" "5579520" "5815721" "5787285" "6134710" "6223337" "6223337" "6249906" "6314562" "6745384" "6944852" "20020049965" "6058265" "6287765" "5754855" "5182806" "5193191" "5201050" "5301327" "5313387" "5325531" "5371747" "5713010" "5812850" "6059839" "6059839" "6260190" "6286135" "5966539" "5999737" "5850554" "6091897" "4980821" "5418959" "5586020" "5734908" "5815719" "5881291" "5999738" "6006033" "6202204" "6202205" "6651246" "6760907" "20020092002" "20020104076" "20040221281" "20050044539" "20050071831").pn.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2006/11/13 14:34
S2	1	10/644619	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2006/11/03 15:58
S3	7	"6240547".pn. "6105148".pn. "6067641".pn. "6044457".pn. "5995754".pn. "5956479".pn. "5933635".pn.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2006/11/03 16:20
S4	14741	compil$6 and optimi$5 and version	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2006/11/03 19:26
S5	137	compil$6 with optimi$5 with version	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2006/11/03 19:26
S6	59	compil$6 with optimi$5 with version and (check$point debug$4)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2006/11/03 19:28

PORTAL
USPTO

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ◉ The ACM Digital Library ○ The Guide

optimized compiling program code and checkpoint SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used
optimized compiling program code and **checkpoint**

Found **72,377** of **192,172**

Sort results by	relevance	Save results to a Binder
Display results	expanded form	Search Tips
		☐ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 Automatic program specialization for Java
Ulrik P. Schultz, Julia L. Lawall, Charles Consel
July 2003 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 25 Issue 4
Publisher: ACM Press
Full text available: pdf(1.18 MB) Additional Information: full citation, abstract, references, citings, index terms

The object-oriented style of programming facilitates program adaptation and enhances program genericness, but at the expense of efficiency. We demonstrate experimentally that state-of-the-art Java compilers fail to compensate for the use of object-oriented abstractions in the implementation of generic programs, and that program specialization can eliminate a significant portion of these overheads. We present an automatic program specializer for Java, illustrate its use through detailed case stud ...

Keywords: Automatic program specialization, Java, object-oriented languages, optimization, partial evaluation

2 Automated application-level checkpointing of MPI programs
Greg Bronevetsky, Daniel Marques, Keshav Pingali, Paul Stodghill
June 2003 **ACM SIGPLAN Notices , Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming PPoPP '03**, Volume 38 Issue 10
Publisher: ACM Press
Full text available: pdf(130.79 KB) Additional Information: full citation, abstract, references, citings, index terms

The running times of many computational science applications, such as protein-folding using *ab initio* methods, are much longer than the mean-time-to-failure of high-performance computing platforms. To run to completion, therefore, these applications must tolerate hardware failures.In this paper, we focus on the stopping failure model in which a faulty process hangs and stops responding to the rest of the system. We argue that tolerating such faults is best done by an approach called appl ...

Keywords: MPI, application-level checkpointing, fault-tolerance, non-FIFO communication, scientific computing

3 Implementation and Evaluation of a Scalable Application-Level Checkpoint-Recovery Scheme for MPI Programs
Martin Schulz, Greg Bronevetsky, Rohit Fernandes, Daniel Marques, Keshav Pingali, Paul

Stodghill
November 2004 **Proceedings of the 2004 ACM/IEEE conference on Supercomputing**
Publisher: IEEE Computer Society
Full text available: pdf(183.27 KB) Additional Information: full citation, abstract

The running times of many computational science applications are much longer than the mean-time-to-failure of current high-performance computing platforms. To run to completion, such applications must tolerate hardware failures. Checkpoint-and-restart (CPR) is the most commonly used scheme for accomplishing this - the state of the computation is saved periodically on stable storage, and when a hardware failure is detected, the computation is restarted from the most recently saved state. Most aut ...

4 FORAY-GEN: Automatic Generation of Affine Functions for Memory Optimizations
Ilya Issenin, Nikil Dutt
March 2005 **Proceedings of the conference on Design, Automation and Test in Europe - Volume 2 DATE '05**
Publisher: IEEE Computer Society
Full text available: pdf(150.39 KB) Additional Information: full citation, abstract, index terms

In today's embedded applications a significant portion of energy is spent in the memory subsystem. Several approaches have been proposed to minimize this energy, including the use of scratch pad memories, with many based on static analysis of a program. However, often it is not possible to perform static analysis and optimization of a program's memory access behavior unless the program is specifically written for this purpose. In this paper we introduce the FORAY model of a program that permits ...

5 When to use a compilation service?
Jeffrey Palm, Han Lee, Amer Diwan, J. Eliot B. Moss
June 2002 **ACM SIGPLAN Notices , Proceedings of the joint conference on Languages, compilers and tools for embedded systems: software and compilers for embedded systems LCTES/SCOPES '02**, Volume 37 Issue 7
Publisher: ACM Press
Full text available: pdf(365.49 KB) Additional Information: full citation, abstract, references, index terms

Modern handheld computers are certainly capable of running general purpose applications, such as Java virtual machines. However, short battery life rather than computational capability often limits the usefulness of handheld computers. This paper considers how to reduce the energy consumption of Java applications.Broadly speaking, there are three interleaved steps in running Java programs in a compiled environment: downloading the bytecodes, compiling and possibly optimizing the bytecodes, and r ...

Keywords: Java, distributed compilation, energy efficient compilation

6 Object and native code thread mobility among heterogeneous computers (includes sources)
B. Steensgaard, E. Jul
December 1995 **ACM SIGOPS Operating Systems Review , Proceedings of the fifteenth ACM symposium on Operating systems principles SOSP '95**, Volume 29 Issue 5
Publisher: ACM Press
Full text available: pdf(1.50 MB) Additional Information: full citation, references, citings, index terms

7 A model and compilation strategy for out-of-core data parallel programs
Rajesh Bordawekar, Alok Choudhary, Ken Kennedy, Charles Koelbel, Michael Paleczny
August 1995 **ACM SIGPLAN Notices , Proceedings of the fifth ACM SIGPLAN symposium on Principles and practice of parallel programming PPOPP '95**, Volume 30 Issue 8
Publisher: ACM Press

Full text available: pdf(1.08 MB) Additional Information: full citation, abstract, references, citings, index terms

It is widely acknowledged in high-performance computing circles that parallel input/output needs substantial improvement in order to make scalable computers truly usable. We present a data storage model that allows processors independent access to their own data and a corresponding compilation strategy that integrates data-parallel computation with data distribution for out-of-core problems. Our results compare several communication methods and I/O optimizations using two out-of-core proble ...

8 Profile-Based Dynamic Voltage Scheduling Using Program Checkpoints
A. Azevedo, I. Issenin, R. Cornea, R. Gupta, N. Dutt, A. Veidenbaum, A. Nicolau
March 2002 **Proceedings of the conference on Design, automation and test in Europe**
Publisher: IEEE Computer Society
Full text available: pdf(611.08 KB) Publisher Site Additional Information: full citation, abstract, citings

Dynamic voltage scaling (DVS) is a known effectivemechanism for reducing CPU energy consumption withoutsignificant performance degradation. While a lot of workhas been done on inter-task scheduling algorithms to implementDVS under operating system control, new researchchallenges exist in intra-task DVS techniques under softwareand compiler control. In this paper we introduce anovel intra-task DVS technique under compiler control usingprogram checkpoints. Checkpoints are generated atcompile time ...

9 Debugging standard ML without reverse engineering
Andrew P. Tolmach, Andrew W. Appel
May 1990 **Proceedings of the 1990 ACM conference on LISP and functional programming**
Publisher: ACM Press
Full text available: pdf(1.29 MB) Additional Information: full citation, abstract, references, citings, index terms

We have built a novel and efficient replay debugger for our Standard ML compiler. Debugging facilities are provided by instrumenting the user's source code; this approach, made feasible by ML's safety property, is machine-independent and back-end independent. Replay is practical because ML is normally used functionally, and our compiler uses continuation-passing style; thus most of the program's state can be checkpointed quickly and compactly using call-with-current-continuation. Together, ...

10 The impact of interprocedural analysis and optimization on the design of a software development environment
Keith D. Cooper, Ken Kennedy, Linda Torczon
June 1983 **ACM SIGPLAN Notices , ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 85 symposium on Language issues in programming environments**, Volume 18 , 20 Issue 6 , 7
Publisher: ACM Press
Full text available: pdf(971.84 KB) Additional Information: full citation, abstract, references, citings, index terms

One of the primary goals of the IRn programming environment project is to mount a concerted attack on the problems of performing interprocedural analysis and optimization in a compiler. Few commercial optimizing compilers employ interprocedural techniques because the cost of gathering the requisite information in a traditional compiler is too great. Computing the side effects of a procedure call requires detailed knowledge of the internals of both the called procedure a ...

11 A framework for efficient reuse of binary code in Java
Pramod G. Joisha, Samuel P. Midkiff, Mauricio J. Serrano, Manish Gupta
June 2001 **Proceedings of the 15th international conference on Supercomputing**
Publisher: ACM Press

Full text available: pdf(419.49 KB) Additional Information: full citation, abstract, references, index terms

This paper presents a compilation framework that enables efficient sharing of executable code across distinct Java Virtual Machine (JVM) instances. High-performance JVMs rely on run-time compilation, since static compilation cannot handle many dynamic features of Java. These JVMs suffer from large memory footprints and high startup costs, which are serious problems for embedded devices (such as hand held personal digital assistants and cellular phones) and scalable servers. A recently propose ...

12 Software tools: EXPERT: expedited simulation exploiting program behavior repetition
Wei Liu, Michael C. Huang
June 2004 **Proceedings of the 18th annual international conference on Supercomputing**
Publisher: ACM Press
Full text available: pdf(156.04 KB) Additional Information: full citation, abstract, references, index terms

Studying program behavior is a central component in architectural designs. In this paper, we study and exploit one aspect of program behavior, the behavior repetition, to expedite simulation. Detailed architectural simulation can be long and computationally expensive. Various alternatives are commonly used to simulate a much smaller instruction stream to evaluate design choices: using a reduced input set or simulating only a small window of the instruction stream. In this paper, we propose to re ...

Keywords: behavior repetition, fast simulation, statistical sampling

13 Multithreading I: Master/slave speculative parallelization
Craig Zilles, Gurindar Sohi
November 2002 **Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture**
Publisher: IEEE Computer Society Press
Full text available: pdf(1.31 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

Master/Slave Speculative Parallelization (MSSP) is an execution paradigm for improving the execution rate of sequential programs by parallelizing them speculatively for execution on a multiprocessor. In MSSP, one processor---the master---executes an approximate version of the program to compute selected values that the full program's execution is expected to compute. The master's results are checked by slave processors that execute the original program. This validation is parallelized by cutting ...

14 Special issue on persistent object systems: Orthogonally persistent object systems
Malcolm Atkinson, Ronald Morrison
July 1995 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 4 Issue 3
Publisher: Springer-Verlag New York, Inc.
Full text available: pdf(5.02 MB) Additional Information: full citation, abstract, references, citings

Persistent Application Systems (PASs) are of increasing social and economic importance. They have the potential to be long-lived, concurrently accessed, and consist of large bodies of data and programs. Typical examples of PASs are CAD/CAM systems, office automation, CASE tools, software engineering environments, and patient-care support systems in hospitals. Orthogonally persistent object systems are intended to provide improved support for the design, construction, maintenance, and operation o ...

Keywords: database programming languages, orthogonal persistence, persistent application systems, persistent programming languages

15 Revising old friends: Capriccio: scalable threads for internet services
Rob von Behren, Jeremy Condit, Feng Zhou, George C. Necula, Eric Brewer

October 2003 **Proceedings of the nineteenth ACM symposium on Operating systems principles**
Publisher: ACM Press
Full text available: pdf(312.83 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper presents Capriccio, a scalable thread package for use with high-concurrency servers. While recent work has advocated event-based systems, we believe that thread-based systems can provide a simpler programming model that achieves equivalent or superior performance.By implementing Capriccio as a user-level thread package, we have decoupled the thread package implementation from the underlying operating system. As a result, we can take advantage of cooperative threading, new asynchronous ...

Keywords: blocking graph, dynamic stack growth, linked stack management, resource-aware scheduling, user-level threads

16 Power reduction techniques for microprocessor systems
Vasanth Venkatachalam, Michael Franz
September 2005 **ACM Computing Surveys (CSUR)**, Volume 37 Issue 3
Publisher: ACM Press
Full text available: pdf(602.33 KB) Additional Information: full citation, abstract, references, index terms

Power consumption is a major factor that limits the performance of computers. We survey the "state of the art" in techniques that reduce the total power consumed by a microprocessor system over time. These techniques are applied at various levels ranging from circuits to architectures, architectures to system software, and system software to applications. They also include holistic approaches that will become more important over the next decade. We conclude that power management is a ...

Keywords: Energy dissipation, power reduction

17 Application-level checkpointing for shared memory programs
Greg Bronevetsky, Daniel Marques, Keshav Pingali, Peter Szwed, Martin Schulz
October 2004 **ACM SIGARCH Computer Architecture News , ACM SIGOPS Operating Systems Review , ACM SIGPLAN Notices , Proceedings of the 11th international conference on Architectural support for programming languages and operating systems ASPLOS-XI**, Volume 32 , 38 , 39 Issue 5 , 5 , 11
Publisher: ACM Press
Full text available: pdf(235.77 KB) Additional Information: full citation, abstract, references, index terms

Trends in high-performance computing are making it necessary for long-running applications to tolerate hardware faults. The most commonly used approach is checkpoint and restart (CPR) - the state of the computation is saved periodically on disk, and when a failure occurs, the computation is restarted from the last saved state. At present, it is the responsibility of the programmer to instrument applications for CPR.Our group is investigating the use of compiler technology to instrument codes to ...

Keywords: checkpointing, fault-tolerance, openMP, shared-memory programs

18 Process migration
Dejan S. Milojičić, Fred Douglis, Yves Paindaveine, Richard Wheeler, Songnian Zhou
September 2000 **ACM Computing Surveys (CSUR)**, Volume 32 Issue 3
Publisher: ACM Press
Full text available: pdf(1.24 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Process migration is the act of transferring a process between two machines. It enables dynamic load distribution, fault resilience, eased system administration, and data access

locality. Despite these goals and ongoing research efforts, migration has not achieved widespread use. With the increasing deployment of distributed systems in general, and distributed operating systems in particular, process migration is again receiving more attention in both research and product development. As hi ...

Keywords: distributed operating systems, distributed systems, load distribution, process migration

19 BugNet: Continuously Recording Program Execution for Deterministic Replay Debugging
Satish Narayanasamy, Gilles Pokam, Brad Calder
May 2005 **ACM SIGARCH Computer Architecture News , Proceedings of the 32nd Annual International Symposium on Computer Architecture ISCA '05**, Volume 33 Issue 2
Publisher: IEEE Computer Society, ACM Press
Full text available: pdf(204.07 KB) Additional Information: full citation, abstract, index terms

Significant time is spent by companies trying to reproduce and fix the bugs that occur for released code. To assist developers, we propose the BugNet architecture to continuously record information on production runs. The information collected before the crash of a program can be used by the developers working in their execution environment to deterministically replay the last several million instructions executed before the crash. BugNet is based on the insight that recording the register file ...

20 An out-of-order execution technique for runtime binary translators
Bich C. Le
October 1998 **ACM SIGOPS Operating Systems Review , ACM SIGPLAN Notices , Proceedings of the eighth international conference on Architectural support for programming languages and operating systems ASPLOS-VIII**, Volume 32 , 33 Issue 5 , 11
Publisher: ACM Press
Full text available: pdf(1.04 MB) Additional Information: full citation, abstract, references, citings, index terms

A dynamic translator emulates an instruction set architccturc by translating source instructions to native code during execution. On statically-scheduled hardware, higher performance can potentially be achieved by reordering the translated instructions; however, this is a challenging transformation if the source architecture supports precise exception semantics, and the user-level program is allowed to register exception handlers. This paper presents a software technique which allows a translato ...

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next